Project Development Phase
Model Performance Test

Date	9 November 2023
Team ID	592320
Project Name	Project - Identifying Airline Passenger Satisfaction using Machine Learning
Maximum Marks	10 Marks

Model Performance Testing:

All the performance testing of Machine Learning Model used for this project is listed below

<table>
<tr><th>S.No</th><th>Parameter</th><th>Values</th><th>Screenshot</th></tr>
<tr><td></td><td>Metrics</td><td>Classification Model:
Confusion Matrix - , Accuracy Score- & Classification Report</td><td></td></tr>
<tr><td>1</td><td>i)Decision Tree</td><td>Confusion Matrix:
[[13573 820]
[767 10636]]
Accuracy Score:
Training accuracy: 1.0
Test accuracy:
0.9389051432091161
Classification Report:
<table>
<tr><th></th><th>0</th><th>1</th></tr>
<tr><td>Precision</td><td>0.95</td><td>0.93</td></tr>
<tr><td>Recall</td><td>0.94</td><td>0.93</td></tr>
<tr><td>F1-score</td><td>0.95</td><td>0.95</td></tr>
<tr><td>support</td><td>14573</td><td>11403</td></tr>
</table></td><td>

```
#Decision Tree
def dt(X_train,y_train,X_test,y_test):
  reg3 = DecisionTreeClassifier(criterion='entropy')
  reg3.fit(X_train,y_train)
  print('accuracy')
  print('Training accuracy= ',reg3.score(X_train,y_train))
  print('test accuracy= ',reg3.score(X_test,y_test))
  y_test_pred = reg3.predict(X_test)
  print('Test data confusion matrix: ')
  print(confusion_matrix(y_test,y_test_pred))
  print('Test data classification report : ')
  print(classification_report(y_test,y_test_pred))
```

```
dt(x_train,y_train,x_test,y_test)
```

```
accuracy
Training accuracy=  1.0
test accuracy=  0.9389051432091161
Test data confusion matrix:
[[13753   820]
 [  767 10636]]
Test data classification report :
              precision    recall  f1-score   support

           0       0.95      0.94      0.95     14573
           1       0.93      0.93      0.93     11403

    accuracy                           0.94     25976
   macro avg       0.94      0.94      0.94     25976
weighted avg       0.94      0.94      0.94     25976
```
</td></tr>
<tr><td></td><td>ii)Random Forest</td><td>Confusion Matrix:
[[14240 333]
[744 10659]]

Accuracy Score:
Training accuracy
0.9999903757314444
Test accuracy
0.9585386510625192
Classification Report:
<table>
<tr><th></th><th>0</th><th>1</th></tr>
<tr><td>Precision</td><td>0.95</td><td>0.97</td></tr>
<tr><td>Recall</td><td>0.98</td><td>0.93</td></tr>
<tr><td>F1-score</td><td>0.96</td><td>0.95</td></tr>
<tr><td>support</td><td>14573</td><td>11403</td></tr>
</table></td><td>

```
[ ]  #Random Forest Model
     def RF(X_train,y_train,X_test,y_test):
       reg4 = RandomForestClassifier(criterion='entropy')
       reg4.fit(X_train, y_train)
       print('Accuracy:')
       print('Training accuracy = ',reg4.score(X_train,y_train))
       print('Test accuracy =',reg4.score(X_test,y_test))
       y_test_pred = reg4.predict(X_test)
       print('Test data confusion matrix')
       print(confusion_matrix(y_test,y_test_pred))
       print('Accuracy Score:',accuracy_score(y_test,y_test_pred))
       print('Test data classification report : ',classification_report(y_test,y_test_pred))
       print('Predicting the data')
```
</td></tr>
</table>

```
RF(x_train,y_train,x_test,y_test)
```

```
Accuracy:
Training accuracy =  0.9999903757314444
Test accuracy = 0.9585386510625192
Test data confusion matrix
[[14240   333]
 [  744 10659]]
Accuracy Score: 0.9585386510625192
Test data classification report :                precision    recall  f1-score   support

           0       0.95      0.98      0.96     14573
           1       0.97      0.93      0.95     11403

    accuracy                           0.96     25976
   macro avg       0.96      0.96      0.96     25976
weighted avg       0.96      0.96      0.96     25976

Predicting the data
```

iii)KNN

Confusion Matrix:

[[12121 2452]
[3617 7786]]

Accuracy Score:

Training Accuracy:

0.8497074222359101

Testing Accuracy:

0.7663612565445026

Classification Report:

	0	1
Precision	0.77	0.76
Recall	0.83	0.68
F1- score	0.80	0.72
support	14573	11403

```
#KNN model
def knn(X_train,y_train,X_test,y_test):
  reg = KNeighborsClassifier(n_neighbors=5)
  reg.fit(X_train,y_train)
  print('Accuracy')
  print('Training accuracy =',reg.score(X_train,y_train))
  print('Test accuracy= ',reg.score(X_test,y_test))
  y_test_pred = reg.predict(X_test)
  print('Test data confusion matrix: ')
  print(confusion_matrix(y_test,y_test_pred))
  print('Test data classification_report :')
  print(classification_report(y_test,y_test_pred))
```

```
knn(x_train,y_train,x_test,y_test)
```

```
Accuracy
Training accuracy = 0.8497074222359101
Test accuracy=  0.7663612565445026
Test data confusion matrix:
[[12121  2452]
 [ 3617  7786]]
Test data classification_report :
              precision    recall  f1-score   support

           0       0.77      0.83      0.80     14573
           1       0.76      0.68      0.72     11403

    accuracy                           0.77     25976
   macro avg       0.77      0.76      0.76     25976
weighted avg       0.77      0.77      0.76     25976
```

iv)Logistic Regression

Confusion Matrix:

[[11662 2911]
[2032 9371]]

Accuracy Score:

Training Accuracy:

0.8121342777948876

Testing Accuracy:

0.809708962118879

Classification Report:

	0	1
Precision	0.85	0.76
Recall	0.80	0.82
F1-score	0.83	0.79
Support	14573	11403

```
#Logistic Regression
def lg(X_train,y_train,X_test,y_test):
  reg1 = LogisticRegression()
  reg1.fit(X_train,y_train)
  print('accuracy')
  print('Training accuracy= ',reg1.score(X_train,y_train))
  print('Testing accuracy= ',reg1.score(X_test,y_test))
  y_test_pred = reg1.predict(X_test)
  print('Test data confusion matrix:')
  print(confusion_matrix(y_test,y_test_pred))
  print('Test data classification report: ')
  print(classification_report(y_test,y_test_pred))
```

			`lg(x_train,y_train,x_test,y_test)` accuracy Training accuracy= 0.8121342777948876 Testing accuracy= 0.809708962118879 Test data confusion matrix: [[11662 2911] [2032 9371]] Test data classification report: precision recall f1-score support 0 0.85 0.80 0.83 14573 1 0.76 0.82 0.79 11403 accuracy 0.81 25976 macro avg 0.81 0.81 0.81 25976 weighted avg 0.81 0.81 0.81 25976
OUT OF ALL THE MACHINE LEARNING MODEL ONLY THE RANDOM FOREST GIVES THE BEST PERFORMANCE (TRAINING AND TESTING ACCURACY IS GREATER THAN 95 %) SO RANDOM FOREST CLASSIFER WILL BE USED FOR THIS PROJECT			
2	**Tunning**	**Not Applied**	Since our model is performing very well, there is no need for Tunning the model